Activity: **6.2**
Design Specifications for Software Modules

Responsibility: Project Team

Description: During the Functional Design Stage, a decomposition of the software product requirements resulted in a collection of design entities (or objects). In the System Design Stage, these design entities are grouped into the routines, modules, and programs that need to be developed or acquired as off-the-shelf or reusable software.

Expand the functional design to account for each major software action that must be performed and each data object to be managed. Detail the design to a level such that each program represents a function that a programmer will be able to code.

Procedure: Use the following procedure to design the software module specifications.

- Identify a software program for each action needed to meet each function or data requirement in the Software Requirements Specification and the data dictionary.

- Identify any routines and programs that may be available as reusable code or objects from existing applications or off-the-shelf software. The System Review Inventory System (SRIS) maintained at DOE Headquarters and the Energy Science and Technology Software Center (ESTSC) located at Oak Ridge, Tennessee are recommended sources for identifying reusable software. The ESTSC is the Department's central collection of DOE-supported software packages. The Center also collects software from the Nuclear Regulatory commission and others, and maintains contact with other software centers.

- Identify programs that must be designed and developed (custom-built). Assign a name to each program and object that is functionally meaningful. Identify the system features that will be supported by each program.

- Specify each program interface. Update the data dictionary to reflect all program and object interfaces changed while evolving from the functional to the system design.

- Define and design significant attributes of the programs to be custom-built.

Procedure, continued:

- Expand the program interfaces to include control items needed for design validity (e.g., error and status indicators).

- Combine similar programs and objects. Group the design entities into modules based on closely knit functional relationships. Formulate identification labels for these modules.

- Show dependencies between programs and physical data structures (e.g., files and global tables). Avoid defining a program that not only needs data residing in a file or global table, but also depends on the physical structure or location of data.

- Change the design to eliminate features that reduce maintainability and reusability (i.e., minimize coupling between programs and maximize the cohesion of programs).

Work Product: Document the system design primarily in the form of diagrams. Supplement each diagram with text that summarizes the function (or data) and highlights important performance and design issues.

When using structured design methods, the design diagrams should:

- Depict the software as a top-down set of diagrams showing the control hierarchy of all software programs to be implemented.

- Define the function of each software program.

- Identify data and control interfaces between programs.

- Specify files, records, and global data accessed by each program.

When using object-oriented or data-centered design methods, the design diagrams should:

- Show the data objects to be managed by the software.

- Specify the program functions to be included within each object.

- Identify functional interfaces between objects.

Work Product, continued:

- Specify files and records comprising each object.
- Identify relationships between data files.

Review Process: Conduct structured walkthroughs to assure that the custom-built routines and programs are correctly designed.